# Resolución Práctica 2.5: Servidor concurrente *Echo* usando UDP

### Fichero *“echo_cli_udp_conc.py”*:

```
#!/usr/bin/env python3

import socket, sys

PORT = 50005
MAX_BUF = 1024

if len( sys.argv ) != 2:
      print( "Uso: {} <servidor>".format( sys.argv[0] ) )
      exit( 1 )

dir_serv = (sys.argv[1], PORT)

s = socket.socket( socket.AF_INET, socket.SOCK_DGRAM )

mensaje = input( "Introduce el mensaje que quieres enviar (mensaje vacío para
terminar):\n" )
if not mensaje:
      exit( 0 )
s.sendto( mensaje.encode(), dir_serv )

buf, dir_serv2 = s.recvfrom( MAX_BUF )
print( buf.decode() )

s.connect( dir_serv2 )

while True:
      mensaje = input()
      if not mensaje:
            break
      s.send( mensaje.encode() )
      buf = s.recv( MAX_BUF )
      print( buf.decode() )
s.send( b"" )
s.close()
```

### Fichero *“echo_ser_udp_conc.py”*:

```
#!/usr/bin/env python3

import socket, os, signal, select

PORT = 50005
MAX_BUF = 1024
MAX_WAIT = 120

s = socket.socket( socket.AF_INET, socket.SOCK_DGRAM )

s.bind( ('', PORT) )

signal.signal(signal.SIGCHLD, signal.SIG_IGN)

while True:
      buf, dir_cli = s.recvfrom( MAX_BUF )
      if not buf:
            continue
```

```
      if not os.fork():
            s.close()
            dialogo = socket.socket( socket.AF_INET, socket.SOCK_DGRAM )
            dialogo.connect( dir_cli )
            while buf:
                  dialogo.send( buf )
                  recibido, _, _ = select.select( [ dialogo ], [], [],
MAX_WAIT )
                  if not recibido:
                        print( "Agotado tiempo máximo de espera ({} s). Fin de
la comunicación.".format( MAX_WAIT ) )
                        break
                  buf = dialogo.recv( MAX_BUF )
            dialogo.close()
            exit( 0 )
s.close()
```